Vol. III, No. 6
July/August 1984

satn®

Software Arts Technical Notes from the creators of VisiCalc® and TK!Solver®

CONTENTS	FEATURING	PAGE
An Inside View of Mortgages We develop a model showing the monthly payment and the first year principal and interest portions of a loan at different interest rates. The model also analyzes the costs of a loan with a 25-year term. You can create this model using built-in functions of the VisiCalc Advanced Version program or develop a similar model with other versions of the program.	@MIN @PMT @PV @ROUND	2
Book Reviews Barry Bayer and Joseph Sobel review two recent books: *Controlling Financial Performance for Higher Profits* and *Planning and Budgeting for Higher Profits*, by Jeffrey R. Alves, Dennis P. Curtin, and Anne K. Briggs.		7
From Our Readers A reader offers some refinements to the TABLE TALK article from the January/February 1984 issue.	@NA @ROUND	8
EOQ: To Buy or Not to Buy Inventory management systems often depend on determination of an economic order quantity (EOQ). Here we show how to use EOQ when the cost of an item varies with the quantity ordered.	@SQRT @AND @IF @SUM	10
Related Products		11
When Something Goes Wrong The TK!Solver program's power makes it possible to build complex models. In this article, we tell you how to recognize and correct some problems you might encounter in model building.		12
Index to Volume III		15

AN INSIDE VIEW OF MORTGAGES

The process of securing a mortgage is filled with questions. How many points will you have to pay? Should you get a fixed or a variable rate loan? Is there a prepayment penalty? Only your bank can answer some of those questions. But you can answer many other questions about mortgages for yourself.

For instance, if you are shopping around for competitive interest rates, you can compare what your costs and tax deductions will be at different rates. After determining the amount, interest rate, and term of the mortgage, you can calculate your monthly payment, as well as how much principal and interest you will pay each year. You can also calculate the cumulative principal and interest paid at the end of any year during the term of the mortgage.

In this article we build a mortgage model with a table that shows a comparison of principal and interest costs corresponding to a range of interest rates. If you are shopping around for a mortgage, you can use this table to quickly look at the costs at different rates. We also develop a table that shows the monthly payment and yearly costs of a loan over a given term. To get this information, we use VisiCalc® Advanced Version's @PV, @PMT, and @ROUND functions. If you are not using the Advanced Version of the program, see the sidebar entitled *A LOOK AT THE FORMULAS*.

Recording the Basic Information

One of the first things you want to find out before committing yourself to a mortgage is how much you must pay each month. That depends not only on how much you borrow and how long you take to pay back the loan, but also on the interest rate you are charged. In our model we examine a $50,000 loan with a 25-year term. (You can use different values if you like.) Given these values, we want to determine the costs associated with different interest rates.

Before building a table that shows the comparative mortgage expenses for different interest rates, set up the top of the worksheet with the basic loan information. You will enter some of the information now, and you will enter or calculate the remaining information later. Begin by setting the global order of recalculation to row and the global column width to 12. Enter:

```
/GOR
/GC12
```

Now enter the labels in rows 1 through 7 across columns A and B, as shown in Figure 1.

To display the LOAN AMOUNT in the dollars-and-cents format, enter:

```
>C1:/F$50000
```

The ANNUAL INTEREST RATE will be used later in the model when we do a full analysis of the loan over its 25-year term. For now leave C2 blank.

We are assuming that the mortgage has a 25-year term and that the first payment is due in June (the sixth month). The 1ST PAYMENT DUE could be any month represented by a number from 1 to 12. Enter:

```
>C3:25
>C4:6
```

The TOTAL PERIODS is the number of monthly payments over which you repay the loan. Calculate this value by multiplying the TERM by the number of months in a year. Enter:

```
>C5:12*C3
```

The MONTHLY PAYMENT in C6 will be calculated later when we examine a loan at a specific interest rate. For the table we are about to build showing the costs associated with different interest rates, we will set up a separate column to display the MONTHLY PAYMENT due for each rate. So leave C6 empty for now.

The ASSUMED BRACKET is the level at which you are taxed and will be used to determine the 1st year TAX DEDUCTION in the comparative interest rate table. We have assumed a 33% tax bracket, but you may enter the bracket that applies to you. Set the percent attribute and enter:

```
>C7:/AV%::.33
```

Comparing Rates

You are now ready to create a second window and develop the table with a range of interest rates. To use the

How to Use SATN

The VisiCalc and TK!Solver programs are available on many different personal computers. These machines may have slightly different keyboards. To reduce confusion, SATN will use the symbol ® and the word RETURN synonymously to represent the keys known as ENTER, ENDLINE, and RETURN. The symbol ^ is used to represent the exponentiation operator.

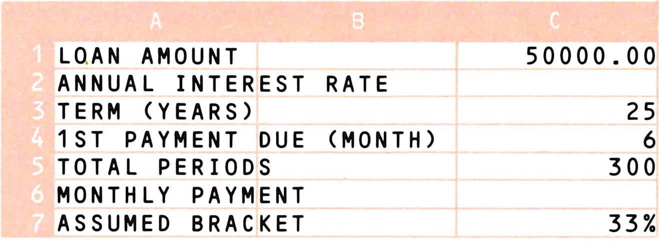
To make VisiCalc documentation more readable, the program was designed so that the colon (:) and the double colon (::) have the same effect as pressing RETURN once or twice in some commands. For instance, the colon is used to end the > command and to separate the various elements of the replicate command. The double colon is used to end the attribute command.

In examples of what to type, type exactly what you see and press RETURN (or two RETURNs if necessary) at the end of every line. On some computers, you may find it necessary to add parentheses to some of the expressions included in the text. For example, 3+4*5 may become (3+4)*5.

Throughout SATN, one typeface represents both VisiCalc and TK!Solver input and output (what you should type and what you will see on the screen). The typeface looks like this:

```
This line is typeset in OCR-B.
So is this: +A1+B1*3.25
```

Characters you should type in are printed in rust.



	A	B	C
1	LOAN AMOUNT		50000.00
2	ANNUAL INTEREST RATE		
3	TERM (YEARS)		25
4	1ST PAYMENT DUE (MONTH)		6
5	TOTAL PERIODS		300
6	MONTHLY PAYMENT		
7	ASSUMED BRACKET		33%

FIGURE 1. *After typing the labels across columns A and B, enter the LOAN AMOUNT, TERM, 1ST PAYMENT DUE, and ASSUMED BRACKET values in column C. Remember to begin the label in A4 with a quotation mark. Then calculate the TOTAL PERIODS by multiplying the TERM by 12. Use the dollars-and-cents format for the LOAN AMOUNT and the percent attribute for the ASSUMED BRACKET. The ANNUAL INTEREST RATE and MONTHLY PAYMENT will be entered later when the complete loan analysis table is built.*

Horizontal option of the Window command, enter:

```
>A8:/WH
```

Next move the cursor to the bottom window by using the semicolon and enter the right-aligned format in rows 8 and 9:

```
>A8:/FR/R:B8...F8
>A8:/R:A9...F9
```

Now enter the labels in A8 through F9 and create a dashed line across row 10, as shown in Figure 2.

In A11 through A19 we will enter the interest rates used to determine the mortgage costs for a $50,000 loan with a 25-year term. We have chosen the range of rates from 12.50% to 14.50%.

Before filling in the rates, set attributes so the values in column A display a percent sign and two decimal places. Enter:

```
>A11:/AV%2F::/R:A12...A19
```

Now enter the first interest rate:

```
>A11:.125
```

Each successive value in A12 through A19 increases by one quarter of a percentage point (.0025), so enter and replicate:

```
>A12:.0025+A11®/R:A13...A19:R
```

In B11 through B19 calculate the monthly payment corresponding to each interest rate. To do that, use @PMT. This function returns the payment required each period to pay off a loan at a specified rate and term. You must supply the interest rate (*i*), number of periods (*n*), and either the present value (*pv*) or the future value (*fv*) of the loan. You can provide both the pv and fv, but if you have only one of these values, you can substitute @NA for the missing value.

To calculate the monthly payment for each interest rate, supply the following arguments:

@PMT(*i*,*n*,*pv*,*fv*)

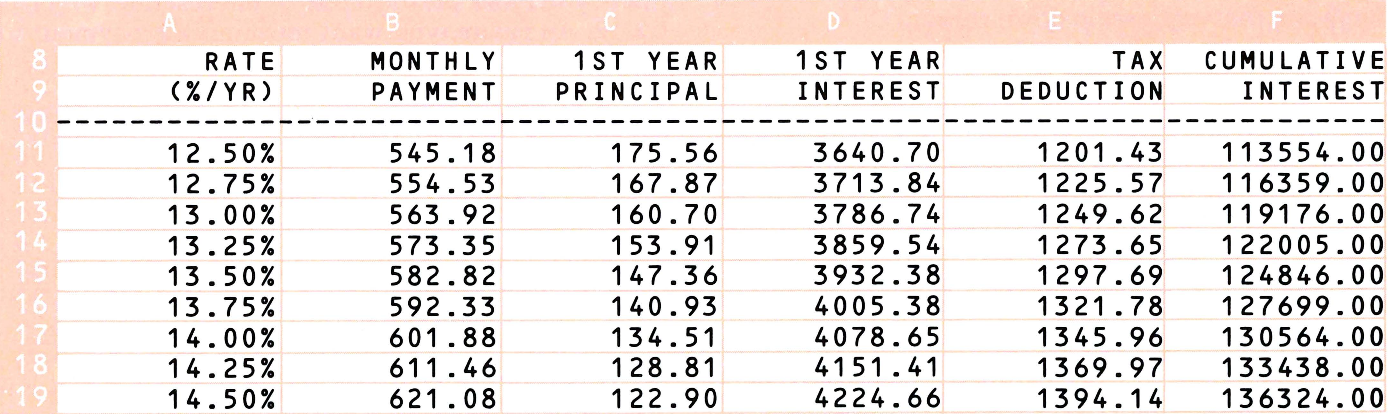
Since you want to know the monthly, not yearly, payment, use a monthly interest rate. The first argument (*i*) of the @PMT function is the rate from column A divided by 12.

The second argument (*n*), the total number of periods, is displayed in the top window in C5.

The third argument (*pv*), present value, should be entered as the negative of the loan amount in C1. By using a negative value, you will have a positive monthly payment. You could enter the present value as a positive amount, but the resulting monthly payment would be negative, which we do not want for this model.

The fourth argument (*fv*), future value, is equal to 0, since after all the payments are made no money is owed.

Banks generally round mortgage payments to two decimal places. Therefore, enclose the @PMT expression within an @ROUND expression and set the precision to 2 decimal places. That will round the MONTHLY PAYMENT to the penny.



	A	B	C	D	E	F
8	RATE	MONTHLY	1ST YEAR	1ST YEAR	TAX	CUMULATIVE
9	(%/YR)	PAYMENT	PRINCIPAL	INTEREST	DEDUCTION	INTEREST
10	------------	------------	------------	------------	------------	------------
11	12.50%	545.18	175.56	3640.70	1201.43	113554.00
12	12.75%	554.53	167.87	3713.84	1225.57	116359.00
13	13.00%	563.92	160.70	3786.74	1249.62	119176.00
14	13.25%	573.35	153.91	3859.54	1273.65	122005.00
15	13.50%	582.82	147.36	3932.38	1297.69	124846.00
16	13.75%	592.33	140.93	4005.38	1321.78	127699.00
17	14.00%	601.88	134.51	4078.65	1345.96	130564.00
18	14.25%	611.46	128.81	4151.41	1369.97	133438.00
19	14.50%	621.08	122.90	4224.66	1394.14	136324.00

FIGURE 2. *The comparative interest table shows the mortgage costs associated with different interest rates. Column A is formatted for percent signs and two decimal places, and columns B through F for dollars and cents. Remember to begin the labels in A9, C8, and D8 with a quotation mark.*

Now set the dollars-and-cents format and enter the @PMT expression to calculate the MONTHLY PAYMENT at each interest rate:

```
>B11:/F$@ROUND(@PMT(A11/12,C5,
  -C1,0),2)
>B11:/R:B12...B19:RNN
```

The 1ST YEAR PRINCIPAL is calculated by subtracting the loan balance after the first year from the total amount borrowed. To do so, we have to calculate the unpaid balance of the loan at the end of the first year. In mathematical terms there is no difference between calculating the present value of a series of payments and calculating the unpaid balance on a loan. Therefore we use the @PV function to calculate the mortgage balance after the first year.

To use the @PV function, you must provide the interest rate, the number of periods, and either the payment or the future value. As with @PMT, both the third and fourth arguments (in this case, the payment and future value) can be provided, but if only one is available you can substitute @NA for the missing value. Because we calculated the payment in column B and we know the future value of the loan is 0, all four arguments for the @PV function are available. So use @PV(*i*,*n*,*pmt*,*fv*).

The interest rate is the same as the monthly rate used in the @PMT function in column B. The number of periods equals the TOTAL PERIODS in C5 minus the number of payments made in the first year. You can calculate how many payments are made in the first year by subtracting the 1ST PAYMENT DUE (C4) from 13. The reason for using 13 rather than 12 is to account for the month in which you make the first payment, as well as the remaining months in the year. For instance, when the first payment is due in June, you will make 7 payments in the first year (13–6). For the payment use the negative of the value in the MONTHLY PAYMENT column (since we want the principal to be a positive value). The future value of the loan is 0.

Now enter the dollars-and-cents format and calculate the principal paid in the first year of the mortgage:

```
>C11:/F$+C1-@PV(A11/12,C5-(13-C4),
  -B11,0)
```

Replicate that expression down column C:

```
>C11:/R:C12...C19:NRNNR
```

The 1ST YEAR INTEREST is simply the number of payments made in the first year (13–C4) multiplied by the MONTHLY PAYMENT (in column B) minus the FIRST YEAR PRINCIPAL (in column C). Use the dollars-and-cents format and enter and replicate that expression in column D:

```
>D11:/F$(13-C4)*B11-C11
>D11:/R:D12...D19:NRR
```

The TAX DEDUCTION in the first year depends on both your tax bracket and the interest you have paid in that year. Format the cell for dollars and cents and multiply the value in column D by the tax bracket in C7:

```
>E11:/F$+D11*C7:/R:E12...E19:RN
```

Now that you have calculated your monthly payment and your first year principal, interest, and tax deduction for each interest rate, you may want to know the total interest amount you will pay over the course of the loan. To do so, multiply the MONTHLY PAYMENT (in column B) by the TOTAL PERIODS (in C5) and then subtract the LOAN AMOUNT (in C1). Use the dollars-and-cents format and enter:

```
>F11:/F$+B11*C5-C1:/R:F12...F19:RNN
```

Having completed this table, you now have a somewhat detailed basis of comparison for what a $50,000 mortgage with a 25-year term will cost, depending on the interest rate you are charged.

Year by Year

Assume you have shopped around and found that the lowest interest rate at which you can secure a $50,000 loan with a 25-year term is 14%. In that case you want a breakdown of your yearly principal and interest payments, as well as the cumulative principal and interest paid at the end of each year of the loan.

To build this portion of the mortgage model, begin by moving the cursor to the top window by using the semicolon. Set attributes in C2 so the interest rate will display a percent sign and two decimal places. Enter:

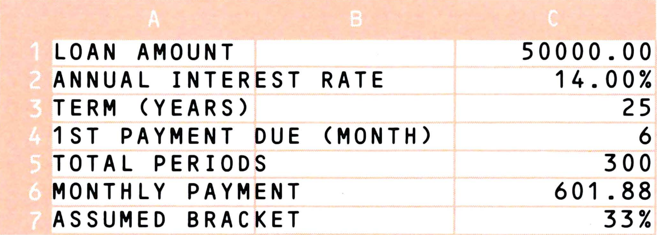
```
>C2:/AV%2F
```

Now enter the annual rate:

```
>C2:.14
```

The MONTHLY PAYMENT in the top window is calculated as it was in the bottom window, using the @PMT and @ROUND functions. The value in C6 is used later to calculate yearly and cumulative principal and interest, so you want the rounded payment used throughout the full term of the loan.

To calculate the MONTHLY PAYMENT in C6, use the same form of the @PMT function as you used in the interest table, providing the interest rate, total number of periods, present value, and future value. Remember that because you are determining the monthly payment, you must use the monthly interest rate, which is C2/12. As before, you want to round the payment to



	A	B	C
1	LOAN AMOUNT		50000.00
2	ANNUAL INTEREST RATE		14.00%
3	TERM (YEARS)		25
4	1ST PAYMENT DUE (MONTH)		6
5	TOTAL PERIODS		300
6	MONTHLY PAYMENT		601.88
7	ASSUMED BRACKET		33%

FIGURE 3. *With the ANNUAL INTEREST RATE entered and the MONTHLY PAYMENT calculated, the top window is complete. The interest rate is formatted for percent signs and two decimal places and the payment is displayed in the dollars-and-cents format.*

two decimal places. Now format C6 for dollars and cents and enter:

```
>C6:/F$@ROUND(@PMT(C2/12,
  C5,-C1,0),2)
```

The resulting value is 601.88, the same as in B17.

Now that the top window is complete for the loan at a 14% interest rate, as shown in Figure 3, you are ready to develop a table to analyze your yearly payments. Put this table in the bottom window to the right of the interest rate table, beginning in column N. (You can begin it in a different column if you like, as long as it is below the information in the top window and does not interfere with other parts of the model.)

Move the cursor to the bottom window and scroll over to column N. Make columns N and O eight characters wide and enter the right-aligned format in N8 through T9:

```
>N8:/GCC8
>O8:/GCC8
>N8:/FR/R:O8...T8
>N8:/R:N9...T9
```

Now enter the labels in N8 through T9 and create a dashed line across row 10, as shown in Figure 4. Because you will be creating a long table, set the horizontal titles at row 10 so you can scroll through the 25 years of the loan while keeping the column headings in view. Enter:

```
>N10:/TH
```

In the YEAR column you want the numbers 1 through 26. Since only seven payments are made in the first year, the final mortgage payments on a 25-year loan will be made in the twenty-sixth year. Enter:

```
>N11:1
>N12:1+N11:/R:N13...N36:R
```

If you want to analyze a 30-year loan, simply replicate the expressions in this and all the remaining columns in the table down through row 41.

Since the first payment is due in the sixth month (C4), the CUMULATIVE PAYMENTS in the first year equal 13–C4. That expression works regardless of which month your first payment is due. Enter the expression in O11:

```
>O11:13-C4
```

In each successive year, the CUMULATIVE PAYMENTS equal the sum of the number of payments made in the preceding year and 12. However, at no time should the number of payments exceed the TOTAL PERIODS (C5). To ensure that this never happens, whether you use this model to examine a loan over 15, 20, 25, or more years, use the @MIN function. Enter:

```
>O12:@MIN(12+O11,C5)
>O12:/R:O13...O36:RN
```

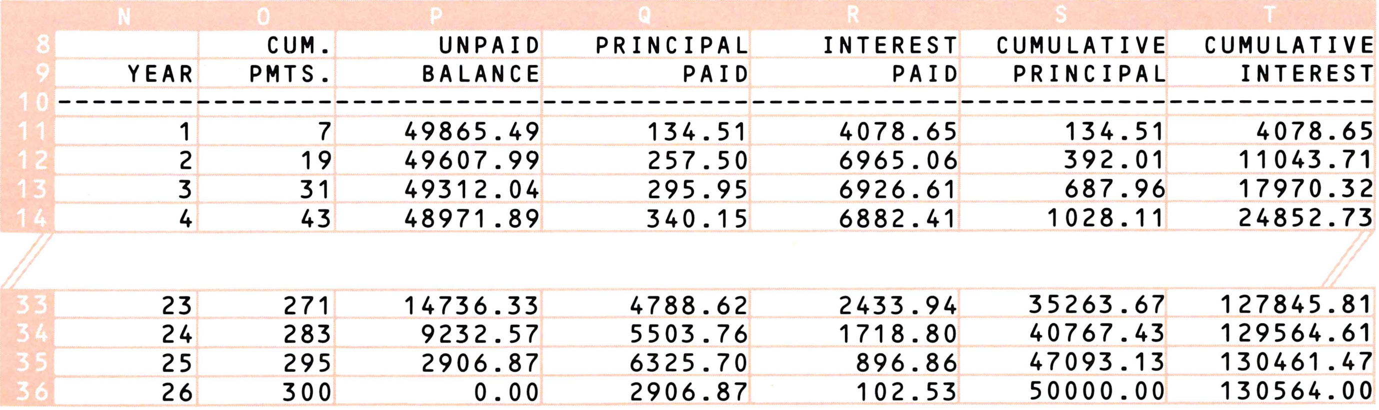
The next five columns will be filled with dollar amounts, so enter and replicate the dollars-and-cents format from P11 through T36:

```
>P11:/F$/R:Q11...T11
>P11:/R...T11:P12...P36
```

To determine the UNPAID BALANCE at the end of any year during the term of the loan, use the @PV function. As when you used the @PV function to calculate the 1ST YEAR PRINCIPAL in the comparative interest rate table, use @PV(*i,n,pmt,fv*) here.

Since you are dealing with monthly payments, use the monthly interest rate (C2/12). The number of periods is the TOTAL PERIODS (C5) minus the CUMULATIVE PAYMENTS as of the year for which you are calculating the balance (column O). For the payment use the negative of the MONTHLY PAYMENT (–C6) so the balance will be positive. The future value is 0.

Since banks usually round amounts to two decimal places, calculate the UNPAID BALANCE by using the @ROUND function with a precision value of 2. Without rounding, the balance would have trailing digits since



	N	O	P	Q	R	S	T
8		CUM.	UNPAID	PRINCIPAL	INTEREST	CUMULATIVE	CUMULATIVE
9	YEAR	PMTS.	BALANCE	PAID	PAID	PRINCIPAL	INTEREST
10	--------	--------	----------	----------	----------	----------	----------
11	1	7	49865.49	134.51	4078.65	134.51	4078.65
12	2	19	49607.99	257.50	6965.06	392.01	11043.71
13	3	31	49312.04	295.95	6926.61	687.96	17970.32
14	4	43	48971.89	340.15	6882.41	1028.11	24852.73
33	23	271	14736.33	4788.62	2433.94	35263.67	127845.81
34	24	283	9232.57	5503.76	1718.80	40767.43	129564.61
35	25	295	2906.87	6325.70	896.86	47093.13	130461.47
36	26	300	0.00	2906.87	102.53	50000.00	130564.00

FIGURE 4. *By scrolling down the loan analysis table, you can see the unpaid balance and the amount of principal and interest paid at the end of any year during the term of the loan. Notice that the final mortgage payments are made in the 26th year. The labels in rows 8 and 9 are right aligned and the values in columns P through T are formatted for dollars and cents.*

the monthly interest rate used in calculating the balance is .011667 (.14/12).

Now enter and replicate the expression for the UNPAID BALANCE:

```
>P11:@ROUND(@PV(C2/12,C5-O11,
  -C6,0),2)
>P11:/R:P12...P36:NNRN
```

Now that you know the balance remaining on the loan, calculating the PRINCIPAL PAID in the first year is easy. Simply subtract the first year's UNPAID BALANCE from the LOAN AMOUNT. Enter:

```
>Q11:+C1-P11
```

The PRINCIPAL PAID in subsequent years requires a similar expression, but you must subtract from it the sum of all the principal paid in preceding years. Use the @SUM function with an expression like the one for the first year principal and enter:

```
>Q12:+C1-P12-@SUM(Q11...Q11)
```

Now replicate the expression to calculate the running total by holding the beginning of the range (the first Q11) constant and the end of the range (the second Q11) relative.

```
>Q12:/R:Q13...Q36:NRNR
```

Calculate the INTEREST PAID in any year by multiplying the MONTHLY PAYMENT by the number of payments in the year and then subtracting the PRINCIPAL PAID in that year. For the first year enter:

```
>R11:+C6*O11-Q11
```

For the INTEREST PAID in the remaining years of the loan, calculate the number of payments in each year by subtracting the previous year's CUMULATIVE PAYMENTS from the number of payments in the current year. That number is always 12, except in the first and last year of the loan when it may be less. For instance, if your first payment were due in June (as in this model), you would make seven payments in the first year and five in the last.

Now enter the expression for INTEREST PAID in each year after the first year:

```
>R12:+C6*(O12-O11)-Q12
>R12:/R:R13...R36:NRRR
```

The final two columns of this table provide you with an overview of how much cumulative principal and interest you will have paid at the end of any year during the term of the mortgage. The first year's CUMULATIVE PRINCIPAL is the same as the PRINCIPAL PAID that year, so enter:

```
>S11:+Q11
```

For each of the remaining years, the CUMULATIVE PRINCIPAL is the sum of PRINCIPAL PAID in the current year and the previous year's CUMULATIVE PRINCIPAL. Enter:

```
>S12:+Q12+S11:/R:S13...S36:RR
```

The CUMULATIVE INTEREST is calculated in the same way as the CUMULATIVE PRINCIPAL. For the first year of the loan, the cumulative amount is the same as the interest paid that year. In each of the following years, it is the sum of the INTEREST PAID in the current year and the previous year's CUMULATIVE INTEREST. Enter these expressions in column T:

```
>T11:+R11
>T12:+R12+T11:/R:T13...T36:RR
```

A LOOK AT THE FORMULAS

If you are not using the Advanced Version of the VisiCalc® program, you can build a similar mortgage model without the built-in functions. With the Advanced Version, all you have to do is supply the values for the variables and the program enters these values into the formulas for you. With other versions of the program, you can accomplish the same task by entering the values into the following formulas yourself:

$$\mathit{monthly\ payment} = \mathit{loan\ amount} * (\mathit{monthly\ interest\ rate}/(1-(1/(1+\mathit{monthly\ interest\ rate})\hat{}\,\mathit{total\ periods})))$$

$$\mathit{unpaid\ balance} = \mathit{monthly\ payment} * ((1-(1/(1+\mathit{monthly\ interest\ rate})\hat{}\,\mathit{remaining\ periods}))/\mathit{monthly\ interest\ rate})$$

$$\mathit{rounded\ number} = @\mathit{INT}(\mathit{original\ number} * 100+.5)/100$$

For instance, in column B of the bottom window, instead of using the built-in @ROUND and @PMT functions to calculate the MONTHLY PAYMENT for the interest rate table, you would enter:

```
>B11:@INT(C1*((A11/12)/(1-(1/
   (1+(A11/12))^C5)))*
    100+.5)/100
>B11:/R:B12...B19:NRRN
```

Similarly, to calculate the 1ST YEAR PRINCIPAL in column C in the bottom window, you would enter:

```
>C11:+C1-(B11*(1-(1/(1+(A11/
   12))^(C5-(13-C4))))/
    (A11/12))
>C11:/R:C12...C19:NRRNNR
```

The MONTHLY PAYMENT in the top window would be calculated by entering:

```
>C6:@INT(C1*((C2/12)/(1-(1+(C2/
   12))^C5)))*100+.5)/100
```

For the loan analysis table in the bottom window, you would calculate the UNPAID BALANCE by entering:

```
>P11:@INT(C6*(1-(1/(1+(C2/12))^(C5-
   O11))/(C2/12))*
    100+.5)/100
>P11:/R:P12...P36:NNNRN
```

Moving Around

Now that you have entered and calculated all the information for the model, you can use the Keystroke Memory™ feature to move from table to table within the bottom window. Create a keystroke sequence that places the cursor so the loan analysis table fills the window. Begin the sequence by moving the cursor to T19, the lower right corner of portion of the table that you want to appear in the window. Next, send the cursor back to N8, the upper left corner of the table. Use the /K= command, name the sequence "A," and enter:

```
/K=A>T19^R>N8^R
```

Whenever you type Control-K A (by depressing the CONTROL and K keys simultaneously and then typing A) in the bottom window, the loan analysis table appears. (On the Apple IIe, the command is Control-S A.) To bring the comparative interest rate table into view, set a tab in column A. For convenience in case you want to change the interest rates, put the tab at A11, the location of the first interest rate. Enter:

```
>A11:/ATY
```

Now try moving from table to table by alternately using the macro and the tab.

The Overall Picture

With the comparative interest rate table and the complete loan analysis table set up and easily accessible, a significant amount of information about mortgages is at your fingertips. In the first table, you can see the differences in monthly payments, first-year principal and interest, and cumulative interest resulting from various interest rates. You can also compare what the first-year tax deduction will be at different interest rates. As the economy changes and interest rates fluctuate, you can enter the new rates in the table and quickly see the associated costs.

In the loan analysis table, you have a complete picture of the mortgage costs, including the unpaid balance and the amount of principal and interest paid throughout the loan. You can adapt this table to examine a different loan amount, interest rate, or term by simply entering new values in the appropriate cells in the top window and extending the columns in the bottom window as necessary. The model also lets you look at a partial year's payments by changing the month in which the first payment is due. ☐

DON'T MISS AN ISSUE!

If you haven't already subscribed to SATN, now is the time to do it. A subscription costs $30 for 6 issues ($50 outside USA)—just send a check or money order, payable to Software Arts to:

SATN Subscriptions
P.O. Box 97
Newton, MA 02162

BOOK REVIEWS

Controlling Financial Performance for Higher Profits and **Planning and Budgeting for Higher Profits**

by Jeffrey R. Alves, Dennis P. Curtin, and Anne K. Briggs. Curtin & London, Inc., Somerville, MA, and Van Nostrand Reinhold Company, New York, $14.95 each. (In editions for the IBM PC and Apple computers. Disks for each book are available for $29.95.)

Reviewed by Barry D. Bayer and Joseph J. Sobel

Reviews published in SATN are the opinions of the reviewers, not of Software Arts, and may not be construed as official positions or endorsements by Software Arts.

Controlling Financial Performance ("*Controlling*") and *Planning and Budgeting* ("*Budgeting*") are designed to help you use the VisiCalc® program to explore your company's past and future. Whether you use VisiCalc on your Apple or IBM, the books will take you by the hand and teach you to turn on your computer, design and save VisiCalc files, and use various templates to better manage your company.

Controlling *and* Budgeting *deal with two sides of the same coin—the financial health of your business*

A major portion of each book is devoted to step-by-step instructions for building these templates. At each stage of each template, the reader is told the problem to be solved, the authors' method of solving it, how the solution relates to the rest of the template, what keystrokes to enter, and the expected result after each series of keystrokes. (Susan Marsh, the book's designer, has created a clean, easy to read presentation, with attractive graphics and template illustrations.) Which keystrokes to enter and where are spelled out in detail, and it is quite difficult to get lost in a template. New VisiCalc users might welcome the keystroke entry practice, but experienced users will probably wish to buy the disks, which will save a lot of time and energy.

Both books provide the reader with useful tips about their computers, the VisiCalc program, and general business problems. Each table of contents lists the titles and locations of these tips, which appear in shaded boxes. Since the books are organized around solutions to business problems rather than around the VisiCalc program, the tips may not be as effective as they might have been for a user who is encountering the program for the first time. The authors present some information about the Advanced Version of the VisiCalc program and have adapted each template to this version.

Now let's take a look at the templates.

Budgeting and its related disk include a main template containing formulas to calculate a simple projected cash budget, income statement, and balance sheet. The authors have taken a "top down" approach. Sales, for example, begin with an annual sales target that is divided into monthly amounts using seasonal sales patterns. The same philosophy runs through the entire template. We prefer the "bottom up" approach, where projected sales are a total of the calculated projections of monthly sales. The formulas used are adequate to show the principles involved, but would probably be inadequate for more sophisticated applications. For example, the authors introduce seasonality into the forecast, allowing provision for fluctuations in sales by month based on the high and low points of the sales year. Little attempt is made to show how some accounts vary seasonally, either leading or lagging sales changes.

both books provide the reader with useful tips about their computers, the VisiCalc program, and general business problems

We differ with some of the template's underlying logic. For example, the authors use a fixed figure rather than a formula to represent cash collections in the first few months of the year. In general, we tend to make greater use of "variables" that are later included in formulas that can be replicated.

The technique of placing variable assumptions in a "What If?" column on an individual income or expense line is an interesting technique we have not used. As implemented, however, there is danger of some confusion, as some of the numbers in the "What If?" column are really totals of the 12 months to the right, rather than entered numbers. If the Advanced Version of the program is used, the formulas in the "What If?" column should be protected, lest someone innocently destroy them.

Budgeting includes several supporting templates accompanying the main template. The user customizes these templates for accumulating expense accounts and schedules to arrive at sales and other figures. The templates require entry of total monthly amounts and percents for income or expense categories (presumably derived from past experience).

Controlling and its related disk include a main template containing formulas to compute the ratios of various items to net sales and total assets. It also totals the income statement and balance sheet, and includes numerous formulas to compute operating and financial ratios.

Controlling contains detailed explanations about the meaning and usefulness of the various ratios and other statistics used, and really constitutes a short course on how to read a set of financial statements. This advice alone is undoubtedly worth the purchase price to anyone without a financial background.

Controlling and *Budgeting* deal with two sides of the same coin—the financial health of your business. *Controlling* will help you analyze your operating history; *Budgeting* will help you develop a plan for next year. The template formats from either book are useful, but the set would be much more powerful if the ratios and other statistics from *Controlling* could be transferred with DIF™ into the projections in *Budgeting*. Unfortunately, the templates are sufficiently different so that this cannot be done.

The tips, VisiCalc, business, and financial information (some of which are common to both books) are mostly accurate and sensible. The templates illustrate one approach to a very important use for the VisiCalc program. On the whole, we have no doubt that either of the two books will provide a lot of valuable VisiCalc and accounting instruction to the novice and a lot of food for thought for the expert. ☐

Barry Bayer is an attorney and columnist. Joseph Sobel is a CPA with a large national firm. Both are from Chicago.

FROM OUR READERS

I'm writing with a few comments about the *TABLE TALK* article in your January/February 1984 issue (Vol. III, No. 3).

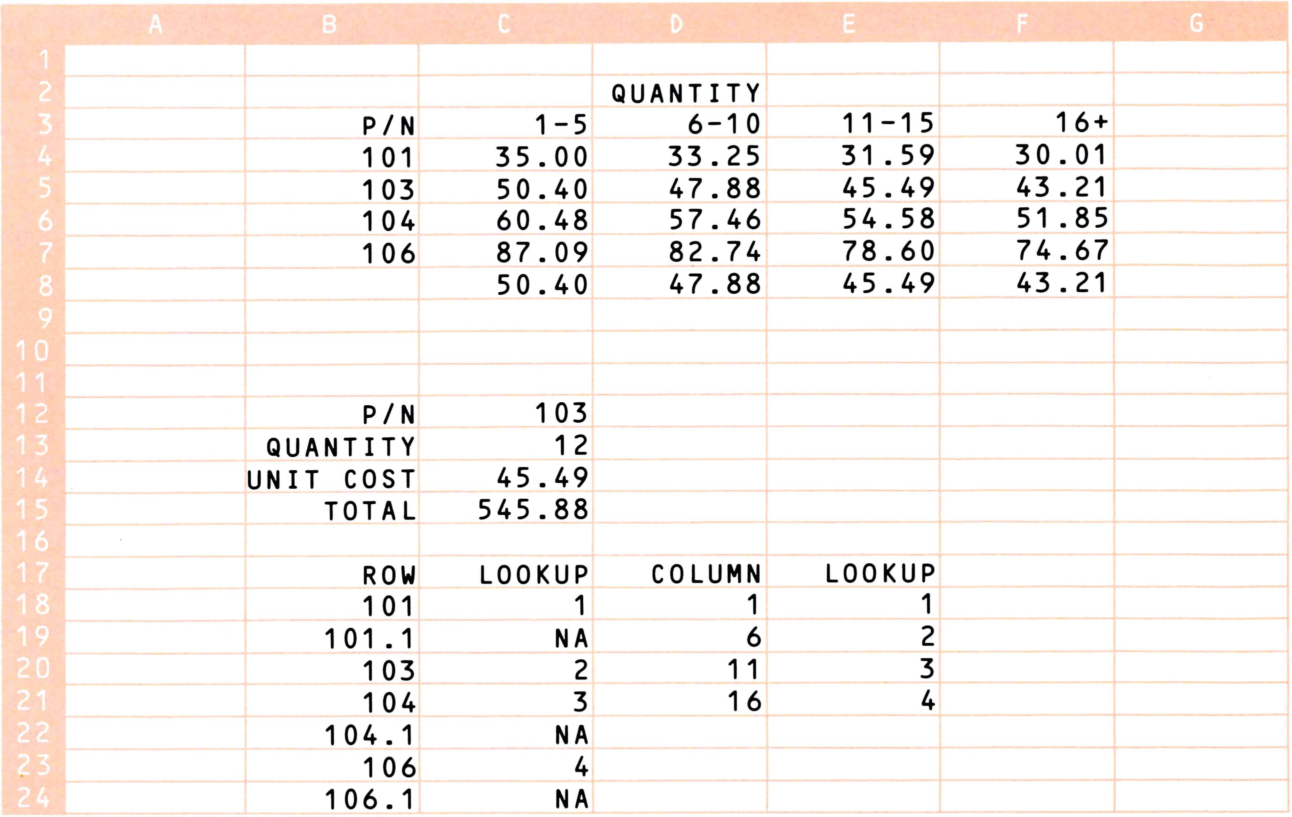
You should point out to your readers that your *TABLE TALK* technique, as you've illustrated it, will not work if the part numbers aren't consecutive. If there's a gap in the part numbers and you enter a number that falls in that gap, you'll get a price based on the next lower part number.

To get around that problem, you have to insert an @NA function into your table to take care of the "in between" numbers. For instance, if part numbers 102 and 105 were missing from a list that ranged from 101 to 106, you should put the intermediate values of 101.1 and 104.1 in the lookup table to take care of the lower end of each gap. Then insert the @NA function opposite each of these values, and you can avoid the problem.

Another problem that I noticed is that when you use a calculated value in your table, you'll have trailing digits after the last two displayed. For example, in your model, 12 parts at $45.49 each total $545.83. The value should be $545.88. Use of the @ROUND function will solve this problem. (Naturally, if the values are entered into the table instead of calculated, you won't have this problem).

These are just minor points. No doubt many of your readers noticed this. Your publication has proved quite valuable to me — keep up the good work.

Gerry A. Young
Financial Strategies N.W.
Eugene, Oregon



	A	B	C	D	E	F	G
1							
2				QUANTITY			
3		P/N	1-5	6-10	11-15	16+	
4		101	35.00	33.25	31.59	30.01	
5		103	50.40	47.88	45.49	43.21	
6		104	60.48	57.46	54.58	51.85	
7		106	87.09	82.74	78.60	74.67	
8			50.40	47.88	45.49	43.21	
9							
10							
11							
12		P/N	103				
13		QUANTITY	12				
14		UNIT COST	45.49				
15		TOTAL	545.88				
16							
17		ROW	LOOKUP	COLUMN	LOOKUP		
18		101	1	1	1		
19		101.1	NA	6	2		
20		103	2	11	3		
21		104	3	16	4		
22		104.1	NA				
23		106	4				
24		106.1	NA				

FIGURE 5. *We have removed part numbers 102 and 105 from the original model (after changing to values all expressions that refer to these rows) and have modified the lookup table to accommodate these gaps. The table now includes the values 101.1 and 104.1 with the @NA function corresponding to each value. If part number 102 or 105 is entered in C12, the UNIT COST and TOTAL, as well as C8 through F8, will return NA. The UNIT COST expression has been edited to include the @ROUND function so the value is rounded to two decimal places.*

Editor's Note: To see how Gerry's suggestion works, you can adapt our original model, as shown in Figure 5. We deleted the rows containing part numbers 102 and 105, but before doing so we used # to change into numbers all expressions that refer to these rows.

In addition to changing the list of part numbers and inserting intermediate values and the @NA function in the lookup table, we have edited the UNIT COST expression to include the @ROUND function. The expression at C14 now reads:

```
@ROUND(@CHOOSE(@LOOKUP(C13,
  D18...D21),C8...F8),2)
```

The UNIT COST will be rounded to two decimal places so there will be no trailing digits beyond what is displayed. Therefore, the QUANTITY multiplied by the UNIT COST will equal the TOTAL (545.88).

If you are not using the Advanced Version of the VisiCalc® program, you can round the UNIT COST with the @INT function. (See Vol. II, No. 6, *Round and Round We Go*, for a full explanation of how to round with @INT.) Use the formula:

rounded number =
@INT(original number ∗ 100 + .5)/100

So the expression at C14 would be:

```
@INT(@CHOOSE(@LOOKUP(C13,D18...D21),
  C8...F8)*100+.5)/100.
```

By the way, we apologize for leaving out of the text of the *TABLE TALK* article two values that were entered in the model. On page 3, the last line ("To serve as examples, enter these values:") should have been followed by:

```
>C12:103
>C13:12
```

□

HAVE YOU MOVED?

If you have, let us know. To keep your issues of SATN coming to you regularly, we need to know your current address. Send a card or letter, indicating both your old address and your new one, to:

SATN Changes
P.O. Box 99
Newton, MA 02162

Remember to allow about 4 weeks for the change to take effect.

EOQ: TO BUY OR NOT TO BUY

In any business, some type of inventory management system is desirable. Generally, these systems weigh carrying cost (cost associated with holding inventory, such as interest, storage, and opportunity costs) against ordering cost. Inventory management systems usually look at the tradeoff between these costs in one of two ways. One is based on ordering a fixed quantity at varying time intervals, and the other is based on ordering a varying quantity at fixed time intervals. We will be looking at a VisiCalc® model that determines the fixed quantity, also known as the economic order quantity (EOQ). Our model will also calculate the reorder point, the actual number of units on hand when more units must be ordered.

For the basic version of the EOQ model to be effective, certain conditions must be met. Throughout the period for which EOQ is being determined (usually one year), the following must be true:

- The demand for the item in question must be constant and uniform;
- The time from ordering the item to receipt of the item (lead time) must be constant;
- The price per unit and the ordering costs cannot change;
- The carrying cost must be based on average inventory; and
- All demand for the item must be satisfied.

Since those criteria cannot always be met, the basic version of the EOQ model isn't always practical. In our model, we will relax certain assumptions, including those of constant and uniform demand and lead time. Since these often are not constant or uniform in real life, many inventory management systems add a buffer to the reorder point, known as safety stock, which our model will also incorporate. This buffer guards against running out of inventory before an order arrives. We will also relax the assumption of a fixed price per unit. Our model will allow for the common occurrence of cost per unit varying with the size of the order.

Suppose that we want to determine the EOQ for an item whose unit cost is as follows: for orders of 1 to 499 units, $5.00; 500 to 1000 units, $4.50; and more than 1000 units, $3.90. The ordering cost is $20.00 per order, no matter how many units we order. The number of units demanded per year is 10,000. Carrying cost—the cost to keep one unit in inventory for one year since we are determining EOQ for one year—is 20% of the cost per unit. Lead time is 10 days, and we wish to keep a safety stock of 100 units.

If this were a simple EOQ model (one without safety stock and with only one possible unit cost), we would determine only the EOQ and the reorder point. Our ordering procedure would be to order the EOQ whenever the number of units in inventory reached the reorder point.

When unit cost depends on the number of units ordered, however, it is necessary to determine an EOQ for each unit cost. Then the total cost of each alternative must be calculated. The least costly ordering procedure depends on the total cost of each alternative.

Begin setting up the model by entering the labels shown in Figure 6. Next enter the values in rows 1...5, 7, and 8 in columns B...D.

The next step is to calculate carrying cost in dollars at each of the three unit costs. Enter:

```
>B6:+B5*B2®/R:C6...D6:RR
```

Next, we must calculate the EOQ at each cost per unit. The formula for calculating EOQ is:

$$EOQ = \sqrt{\frac{2 * \mathit{Ordering\ Cost} * \mathit{Units\ Per\ Period}}{\mathit{Carrying\ Cost}}}$$

In our VisiCalc model, this becomes:

```
>B10:@SQRT(2*B3*B4/B6)
>B10:/R:C10...D10:RRR
```

Next we will calculate the reorder point. The formula for the reorder point without safety stock is:

$$\mathit{Reorder\ Point} = \mathit{Lead\ Time} * \frac{\mathit{Units\ Per\ Period}}{365}$$

With safety stock, the formula becomes:

$$\mathit{Reorder\ Point} = \mathit{Lead\ Time} * \frac{\mathit{Units\ Per\ Period}}{365} + \mathit{Safety\ Stock}$$

Since our model has safety stock, we'll use the second formula. Enter:

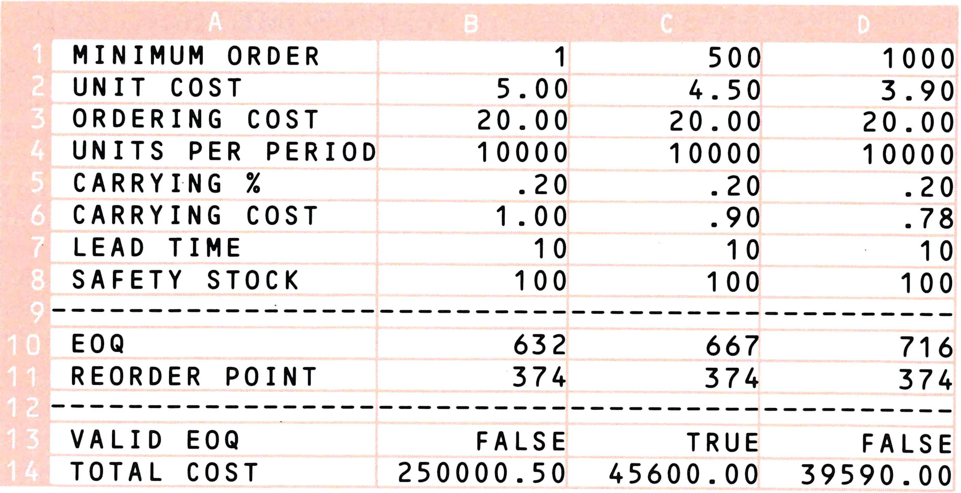
```
>B11:+B7*(B4/365)+B8
>B11:/R:C11...D11:RRR
```

The next step is to determine whether the EOQ at each price per unit is valid. The EOQ is not always within the range of the number of units that can be purchased at a given price. For instance, we can see from cell B10 that the EOQ for a cost per unit of $5.00 is 632. This is not a valid EOQ, since the $5.00 per unit price applies only to quantities less than 500. To discover whether the EOQs in our model are valid, enter:

```
>B13:@AND(B10>=B1,B10<C1)
>B13:/R:C13:RRRR
>D13:+D10>=D1
```

Upon evaluating these expressions, the model will return the value FALSE if the EOQ is not valid and the value TRUE if it is valid.

When an item has several possible costs and several EOQs, the appropriate quantity and unit cost depend on the total cost to meet demand for the period. When the EOQ is valid, the total cost is based on the EOQ. If the EOQ is not valid at a given unit cost, the total cost is based on the minimum acceptable order quantity at that cost. The total costs of ordering a valid quantity at each unit cost are compared. The least costly quantity to order, whether EOQ or minimum quantity at a given



	A	B	C	D
1	MINIMUM ORDER	1	500	1000
2	UNIT COST	5.00	4.50	3.90
3	ORDERING COST	20.00	20.00	20.00
4	UNITS PER PERIOD	10000	10000	10000
5	CARRYING %	.20	.20	.20
6	CARRYING COST	1.00	.90	.78
7	LEAD TIME	10	10	10
8	SAFETY STOCK	100	100	100
9	----------------	----------	----------	----------
10	EOQ	632	667	716
11	REORDER POINT	374	374	374
12	----------------	----------	----------	----------
13	VALID EOQ	FALSE	TRUE	FALSE
14	TOTAL COST	250000.50	45600.00	39590.00

FIGURE 6. *An EOQ model incorporating quantity discounts and safety stock. If you are using VisiCalc Advanced Version, you can set the column widths as they appear here. Columns B, C, and D have a width of 10 (/GC10), and Column A's width is 16 (/GCC16). Other formatting instructions are dollars-and-cents format (/F$) in rows 2, 3, 5, 6, and 14 and integer format (/FI) in rows 10 and 11.*

unit cost, is determined by the unit cost and order quantity that result in the lowest total cost.

The following equation determines total cost:

$$\text{Total Cost} = (\text{Units Per Period} * \text{Unit Cost}) + \left(\frac{\text{Units Per Period}}{\text{Order Quantity}} * \text{Ordering Cost}\right) + \left(\frac{\text{Order Quantity}}{2} * \text{Carrying Cost}\right)$$

If EOQ is valid, then order quantity = EOQ.
If EOQ is invalid, then order quantity = minimum order.

In our VisiCalc model, we want to determine the total cost at each unit cost. Total cost will be based on the EOQ if it is valid or, if the EOQ is not valid, on the minimum possible quantity at that cost. To do this, we will use VisiCalc's @IF function to determine first if the EOQ is valid and the @SUM function to calculate the total cost on the applicable unit cost. (Notice the use of expressions within the @SUM statement and @SUM within the @IF statement.) Enter:

```
>B14:@IF(B13,@SUM(B10/2*B6,
  B4/B10*B3,B4*B2),@SUM(B1/2*B6,
    B4/B1*B3,B4*B2))
```

Now replicate that expression:

```
>B14:/R:C14...D14:RRRRRRRRRRRRRRR
```

Your model should now look like Figure 6, which shows total cost of $250,000.50 at the unit cost of $5.00 and quantity 1, total cost of $45,600 at the unit cost of $4.50 and its associated EOQ of 667, and total cost of $39,590 at the unit cost of $3.90 and associated quantity (minimum possible) of 1000 units. The ordering rule that we can derive from these results is that we should order in quantities of 1000 units, placing an order each time the number of units in inventory falls to the reorder point of 374.

When unit price does not vary and safety stock is not necessary, a simpler model than the one we have developed can be used. Such a model would look like columns A and B to row 11 (minus safety stock in row 8) of our model. When price breaks and safety stock are considerations, however, the EOQ model that we have developed can be quite useful. ☐

RELATED PRODUCTS

No one at Software Arts has reviewed or examined any of these products to determine whether they perform as claimed or as listed below. An inclusion in this listing does not constitute an endorsement by Software Arts of that product. We also wish to point out that if any of the products listed alter, patch, or copy the VisiCalc® program, they could be in violation of your user licensing agreement and/or the copyright.

Since we have not reviewed any of the products, we cannot provide you with any more information than is contained in the list. If you want to learn more or you wish to purchase any of the products listed, contact your dealer first. Although we have identified distributors in some listings, many distributors do not accept orders from individuals for single copies of software or pieces of hardware.

SOFTWARE

Only software that claims to complement the VisiCalc program will be listed.

Templates

Viz-A-Merge

- Runs on Apple II, Apple II+, Apple IIe, Apple ///
- Supports DIF format

Allows you to combine sections, rows, columns, or single cells from any number of VisiCalc worksheet pages into one new report or into an existing report, then recalculate selected formulas. Information can be taken from any part of one worksheet and placed in any part of another. Comes with a question-and-answer procedure to assist you in organizing the report. Reports may be saved in standard ASCII form.

Distributed by Abacus Associates, 6565 W. Loop South, Suite 220, Bellaire, TX 77401

Suggested retail price: $99.95 Apple II version; $139.95 Apple II+, Apple IIe, Apple /// versions

Viz-A-Con

- Runs on Apple II, Apple II+, Apple IIe, Apple ///
- Supports DIF format

Combines a series of VisiCalc worksheets into a single summary by adding the same locations from each file and then storing the sums at the same location in the finished product. Labels of the first file specified for consolidation are what appear on the final version. As many as 50 files may be consolidated at one time. Larger numbers of files may be processed in groups.

Distributed by Abacus Associates, 6565 W. Loop South, Suite 220, Bellaire, TX 77401

Suggested retail price: $99.95 Apple II version; $139.95 Apple II+, Apple IIe, Apple /// versions

F.A.S.T. (Financial Analysis Statement Templates)

- Runs on Apple II, Apple IIe, and IBM PC
- Requires 48K of RAM on the Apple II and Apple IIe; 128K of RAM, two double-sided disk drives, and one display card on the IBM PC

Converts VisiCalc data into reports and comparisons for analysis of a variety of financial documents.

Distributed by Continental Software, 11223 So. Hindry Ave., Los Angeles, CA 90045

Suggested retail price: $99.95

Other Software

Iowa State University Software Distribution Office

- Apple II, Apple IIe and Apple ///

A software distribution office set up for getting programs developed by the university's research and extension personnel into the hands of potential users. The initial software available includes VisiCalc templates.

Distributed by Kevin Gamble, Manager of Software Distribution, 108 Computation Center, Iowa State University, Ames, IA 50011

Suggested retail price: $15.00 diskettes; $2.50 documentation only ☐

WHEN SOMETHING GOES WRONG

The TK!Solver® program is a powerful tool that allows its user to build complex models. The potential for complexity in the model-building process means that a wide variety of things can go wrong at various stages in model development. Because everyone's models are different, there cannot be a definitive "Troubleshooter's Cookbook." That's no consolation when **you** are stumped, so we'll take a look at some common problems you might encounter when you're building a model, and some ways to correct those problems.

Two of the most common problems in model building are entering data incorrectly and misusing commands. When the program beeps, indicating an error while you're entering data or commands, your best bet is the TK!Solver Reference Manual, where you'll find details of each command and the kinds of entries you can make in the various fields.

Solution errors can be far more frustrating and harder to diagnose. These errors occur after you have built a model and are trying to solve it. When a solution error occurs, the program beeps and displays a "greater-than" symbol (>) in the Status column of the Rule Sheet, next to the errant rule, and next to the affected variables from that rule in the Status column of the Variable Sheet.

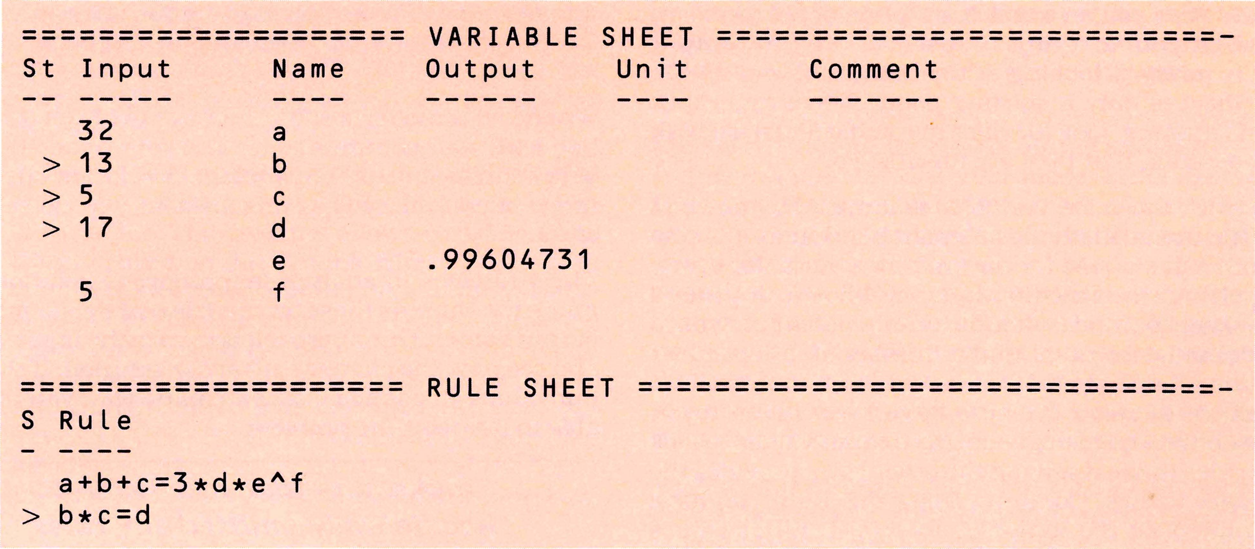
Dealing with Solution Errors

The first step in correcting a solution error is to ask the program what went wrong. Move the cursor to the Status field containing the >; the Prompt/Error line at the top of the screen will display an error message. In the Appendices section of the Reference Manual you'll find a list of the program's Error Messages, with brief discussions of what causes them and what actions you can take to correct the problem.

After you determine the specific type of error, take steps to figure out why the problem occurred. First try varying the input values to see if the problem is restricted to a certain range of values.

pick apart expressions that seem to be causing problems

Sometimes it helps to use the Calculator feature in an empty numeric field (such as the Input field of an unused row of the Variable Sheet) to pick apart expressions that seem to be causing problems on the Rule Sheet. For example, say a rule is marked in error. Copy it into an unused Comment field on the Variable Sheet and edit it so that only part of the expression remains. Then copy that part into an unused Input field and see if it can be evaluated. If it is evaluated correctly, repeat the process with other pieces of the expression and see what happens. Gradually, you should isolate the part of the rule that is causing the problem.



```
==================== VARIABLE SHEET ===========================-
St Input      Name     Output     Unit        Comment
-- -----      ----     ------     ----        -------
    32         a
  > 13         b
  > 5          c
  > 17         d
               e      .99604731
    5          f

==================== RULE SHEET ================================-
S Rule
- ----
  a+b+c=3*d*e^f
> b*c=d
```

FIGURE 7. *The Variable and Rule Sheets of a model with the problem of overdefinition. The input values provided for* b, c, *and* d *cause a contradiction when TK!Solver tries to solve the second equation on the Rule Sheet.*

Solution errors can also result from giving too much or contradictory information on the Variable Sheet. Figure 7 shows the Variable and Rule Sheets of a model that was given contradictory information. When you move the cursor to the > in the Status column of the Rule Sheet next to the rule $b*c=d$, the message > **Inconsistent** is displayed on the Prompt/Error line. Moving the cursor to the > in the Variable Sheet's Status column for the variables b, c, or d causes the message > **Overdefined** to be displayed.

One way to address the problem of contradictory information is to blank the overdefined input values one at a time and solve each time you blank one value. This process may show which input values are causing the problem. In the model shown in Figure 7, blanking the input value for any of the variables b, c, or d will result in a solution of the equation $b*c=d$.

Another way to address the problem is to add another variable to the faulty expression. That will tell you the magnitude of the error. For example, if you edit the expression $b*c=d$ to read $b*c=d+error$, the value 48 will be returned for *error*. Now you know that $b*c$ is 48 less than d. That constitutes a sizable error in this problem since the largest variable in the problem has a value of 32. In a problem whose smallest variable was 100 million, however, an error of 48 would be minimal.

If you defined the relationship $a=b$ and entered input values of 100,000,000 for a and 100,000,048 for b, TK!Solver would not call the expression inconsistent. TK!Solver sees 100,000,000 and 100,000,048 as equal because of the setting of the comparison tolerance on the Global Sheet. Changing the comparison tolerance can eliminate inconsistencies, but it should be done carefully. If you are working with variables of very different sizes, .075 and 300,000, for example, changing the comparison tolerance can cause problems, which we'll discuss in a future issue of SATN.

Inconsistencies that arise from giving the program contradictory information can be very subtle, and the input variables marked with a > in the Status column may not be the precise source of your difficulty. It also pays to compare the rules against the source from which they were taken to be sure no typing errors have occurred.

blank the overdefined input values one at a time

There are other strategies to investigate models that give contradictions. Try placing a double quote (") in front of what seems to be the offending rule. That turns the rule into a comment, removing it from the solution process. See if the model works with the rule "commented out." Since TK!Solver stops trying to solve a model when it runs into an error, you may find that the first rule you turn into a comment is not the only problem. Repeating the process of turning rules into comments, then trying to solve the model can pinpoint the rules that are causing your problems.

Unsatisfactory Results

Even if no solution error has occurred, a model may not be working properly. Here we'll describe ways to track down and correct problems when your model gives unsatisfactory results by returning no answer or an unreasonable answer.

If solving the model does not give you output values for the unknowns you seek, the first thing to do is to make sure that you haven't accidentally omitted any rules from the Rule Sheet.

If no rules have been omitted, you should look at the Status column of the Rule Sheet as you do with solution errors. When you have a solution error, the Status column gives you an exact description of the problem, but when you're trying to discover why an output value is missing, looking at the Status column on the Rule Sheet is only a starting point. When an output value is missing, look for rules marked with an asterisk (*), indicating that they are unsatisfied.

Next, track down the variables that must be known to solve the unsatisfied rules. Remember that for the program to solve a rule for an unknown variable, everything else in the equation must be known. A general rule to keep in mind is that the number of unknowns in a model must be equal to the number of independent equations. Look at each rule to see if all the variables necessary to solve the rule have been given input values. If they have not been given input values, check other rules to see if solving the other rules provides the necessary values. Continue doing this until you have determined all the variables for which input values must be entered.

Figure 8 shows an example of a model with an unsatisfied rule. The rule $g=e+f$ is marked with an *, indicating that it is not solved. We do not have to go very far in checking whether input values have been given for all the variables necessary to solve for g. Neither e nor f has an input value, but the value of e is found by the equation $b=(c+d)*e$. No equations give us a value for f, so we know that $g=e+f$ is not solved because f has no value associated with it.

There are several possible causes of an unsatisfied rule. If an unknown variable occurs more than once in a rule, then the model must be solved iteratively to find a value for that unknown. (We'll discuss troubleshooting for models using iterative solving in the next issue of SATN.) Another reason a rule may not be solved is that the TK!Solver program can't backsolve past uninvertible functions like MAX, INT, and STEP. Finally, an equation may not be solved because of the presence of an indeterminate form such as 0/0 or 0^0. The TK!Solver program will not evaluate a rule containing one of those indeterminate forms. (For further explanation of indeterminate forms, please see *Rule Switching* in TK!SATN™, Vol. 1, No. 4.)

Whenever a model is solved, it pays to check the solution with your common sense and your understanding of the constraints of the problem. A solution out of the range of possibility is as unsatisfactory as no solution at all.

The strategies to analyze inappropriate solutions are much the same as those for models that do not produce output values. One approach is to vary the input values and see how the changes affect the solution. If the output values all change in a systematic way, you may be able to pinpoint the problem.

make sure that you haven't accidentally omitted any rules

The TK!Solver program's backsolving ability can be put to good use in models that give unreasonable answers. Enter a reasonable value for one of the unreasonable output variables, blank one of the input values, and solve. By solving for what is normally an input value, you may find that the relationships in the model are different from those you expected.

Just as errors in deriving or entering an equation can result in missing output values, such errors can cause inappropriate solutions. Check your derivations and check the Rule Sheet against the references you used. Sign errors (confusion of plus and minus signs) are probably the most common derivation and copying errors. Again, you can isolate parts of the rules by using the Calculator feature in unused Input fields. (Remember that the Calculator takes all variable values in Calculation units.)

Problems with precedence of operations are also common. The program may not be performing arithmetic operations in the order you think it is. Unless you tell it

```
==================== VARIABLE SHEET ===========================-
St Input     Name    Output    Unit      Comment
-- -----     ----    ------    ----      -------
   400       a
   125       b
             c       275
   90        d
             e       .34246575
             g
             f

==================== RULE SHEET ==============================-
S Rule
- ----
  a=b+c
  b=(c+d)*e
* g=e+f
```

FIGURE 8. *The Variable and Rule Sheets of a model with unsatisfactory results because no output value was found for* g.

to do otherwise, TK!Solver will perform exponentiation (^) first, followed by multiplication (∗) and division (/), and finally, addition (+) and subtraction (−). To make operations take place in another order, use parentheses within the equation.

Perhaps the subtlest errors resulting in unreasonable answers are those that relate to TK!Solver's use of two forms of variables. The program allows variables to be entered and viewed on the Variable Sheet in one form (Display unit) while it uses another form of the variable in its calculations (Calculation unit). Both Calculation and Display units would be used in a model in which you enter values in feet, but the model's equations use inches. You might enter 5 as an input variable, but TK!Solver would use 60 as the Calculation unit. The steps that the program must take to convert the 5 you enter into the 60 it uses must be defined on the Unit Sheet.

When your model produces unreasonable answers, make sure that all the Calculation units are consistent. Check, too, that all the rules use the appropriate units and use them consistently. Since Unit Sheet errors can affect answers greatly, also check the conversions on the Unit Sheet. One common error is to inadvertently transpose the *From* and *To* columns on the Unit Sheet. It clearly makes a difference if your model thinks that 1 inch = 12 feet instead of 1 foot = 12 inches.

If you've tried all the approaches we've suggested and still haven't found the reason for the unsatisfactory solution, it is possible that your model is not appropriate for the problem you want it to solve. Models are based on assumptions, and assumptions that are valid for one problem may not be valid for another problem that seems similar. For example, a model that determines the proper load for a wooden beam might include assumptions that make it inappropriate for determining the load on a steel beam. Make sure that the problem you are asking the model to solve is one in which all of the assumptions can reasonably be expected to hold.

TK!Solver is a powerful tool, but with all its complexity, it must be provided with a logically sound model to produce a satisfactory solution. If you run into problems with a model, there are several things you can do. Using the approaches we have suggested here can be most helpful as can the program's Help Facility and the TK!Solver Reference Manual. If all else fails, you can reach Software Arts' technical support for TK!Solver by calling 617-431-6500. □

INDEX TO VOLUME III

[AV] refers to specific mentions of functions or commands in the VisiCalc Advanced Version program.

^< #4 p.8 [AV]
^> #4 p.8 [AV]
^D #4 p.8 [AV]
^K #4 p.8 [AV]
^P #4 p.8 [AV]
^R #4 p.8 [AV]
^U #4 p.8 [AV]
/- #3 p.6
/A #3 p.6 [AV]
/AE #3 p.5 [AV]
/AH #5 p.5 [AV]
/F∗ #3 p.3
/F= #3 p.6 [AV]
/G #5 p.6
/GCC #3 p.6 [AV]
/K #4 p.7 [AV]
/L! #5 p.8 [TK!Solver]
/PF #2 p.6; #4 p.4
/R() #1 p.3 [AV]
/S#L #3 p.11
/S#S #3 p.11
/T #5 p.6
/W #5 p.7
@AND #6 p.10
@CHOOSE #3 p.2
@COUNT #1 p.3
@DOTPROD #1 p.3; #5 p.2 [AV]
@EXP #1 p.3
@IF #2 p.3; #5 p.4; #6 p.11
@INT #5 p.2
@IRR #4 p.2 [AV]
@ISERROR #2 p.3
@LABEL #3 p.6; #5 p.2 [AV]
@LCHOOSE #5 p.2 [AV]

SATN is published bimonthly by Software Arts Products Corp., 27 Mica Lane, Wellesley, MA 02181. Address all editorial correspondence to the Editorial Department, SATN, P.O. Box 99, Newton, MA 02162. Requests for special permission or bulk orders should be addressed to the publisher.

Domestic subscriptions: 6 issues $30.00, 12 issues $50.00; outside U.S.A., 6 issues $50.00, 12 issues $90.00. Single-copy price, $6.00. Back-issue price, $7.00. Send subscriptions, change of address, and fulfillment questions to SATN Subscriptions, P.O. Box 97, Newton, MA 02162.

Copyright © 1984 Software Arts Products Corp. All rights reserved. This periodical may not be copied or reproduced in any fashion either in whole or in part without the prior written consent of Software Arts Products Corp.

Software Arts Products Corp. makes no warranty, express or implied, with respect to the quality, accuracy, or freedom from error of the information contained in SATN, and specifically disclaims all liability for any damages resulting from the use of such information.

Unless specifically stated otherwise in any article in this issue of SATN, Software Arts Products Corp. has not examined any products and/or publications identified in SATN to determine their accuracy and/or ability to perform as advertised. Mention of a product and/or publication in SATN does not constitute an endorsement by Software Arts Products Corp. of any such product or publication.

Recognition in SATN of the existence of any bug or problem in any product distributed by Software Arts, Inc. does not constitute, and should not be construed as, an acknowledgment by Software Arts, Inc. of any deficiency or as a commitment by Software Arts, Inc. to correct the bug or problem.

- @LN #1 p.3
- @LOOKUP #2 p.8; #3 p.2; #5 p.2; #5 p.10
- @MAX #5 p.2
- @MIN #6 p.5
- @NA #6 p.8
- @NPV #4 p.3
- @PMT #6 p.3 [AV]
- @PV #6 p. 5 [AV]
- @RATE #4 p.4 [AV]
- @ROUND #3 p.6; #6 p.3,8
- @SQRT #1 p.6; #6 p.10
- @SUM #1 p.3; #2 p.4; #6 p.11
- Adjusted Internal Rate of Return (AIRR) #4 p.2
- Adjusted Net Present Value (ANPV) #4 p.2
- Argument Error #5 p.8 [TK!Solver]
- Books Reviewed
  - Alves, J.R.; Curtin, D.P.; and Briggs, A.K.: *Controlling Financial Performance for Higher Profits* #6 p.7
  - Alves, J.R.; Curtin, D.P.; and Briggs, A.K.: *Planning and Budgeting for Higher Profits* #6 p.7
  - Bayer, B.D. and Sobel, J.J.: *Dynamics of VisiCalc* #4 p.10
  - Desautels, E.J.: *Advanced Uses of VisiCalc for the IBM Personal Computer* #1 p.9
  - Hergert, D.: *Mastering VisiCalc* #3 p.10
  - Klitzner, C. and Plociak, M.J.: *Using VisiCalc: Getting Down to Business* #3 p.10
  - Nevison, J.: *The Executive Package - Computing for the Corporate Manager with VisiCalc and BASIC* #2 p.9
- Circular References #2 p.2
- Control-K= command #4 p.7 [AV]
- Control-S= command #4 p.7 [AV]
- Correlation Coefficient #1 p.2
- DIF Pointer #3 p.11
- Economic Order Quantity (EOQ) #6 p. 10
- ELEMENT #5 p.8 [TK!Solver]
- Forward References #2 p.2
- General Formula for:
  - Adjusted Internal Rate of Return (AIRR) #4 p.3
  - Adjusted Net Present Value (ANPV) #4 p.3
  - Correlation Coefficient #1 p.3
  - Economic Order Quantity (EOQ) #6 p.10
  - Intercept #1 p.6
  - Projected Value #1 p.7
  - Slope #1 p.6
- Intercept #1 p.6
- Keystroke Memory #4 p.7 [AV]
- Least Squares Fitting #1 p.3
- Linear Regression #1 p.2
- List #5 p.8 [TK!Solver]
- Lower Right Corner of Model #3 p.5
- Macros within Macros #4 p.9 [AV]
- NPV #5 p.8 [TK!Solver]
- On/Off Switch #5 p.3
- Pointers
  - Calculating with the # and ! Keys #4 p.9
  - Expanding Your @LOOKUP Range #2 p.8
  - Giving Tables a DIFferent Look #3 p.11
- POLY #5 p.8 [TK!Solver]
- Products Reviewed
  - Accelerator II, Titan Technologies, Inc., #1 p.8
  - ScreenWriter II, Sierra On-Line, Inc., #2 p.6
- Projected Value #1 p.7
- Slope #1 p.6
- Sorting #4 p.4
- SQRT #5 p.8 [TK!Solver]
- Tables #3 p.11; #6 p.8
- Terminal Value #4 p.3
- TK!Solver Commands and Functions
  - /L! #5 p.8
  - ELEMENT #5 p.8
  - NPV #5 p.8
  - POLY #5 p.8
  - SQRT #5 p.8
  - Argument Error #5 p.8
  - List #5 p.8
- VisiCalc Advanced Version Commands and Functions
  - ^< #4 p.8
  - ^> #4 p.8
  - ^D #4 p.8
  - ^K #4 p.8
  - ^P #4 p.8
  - ^R #4 p.8
  - ^U #4 p.8
  - /A #3 p.6
  - /AE #3 p.5
  - /AH #5 p.5
  - /F= #3 p.6
  - /GCC #3 p.6
  - /K #4 p.7
  - /R() #1 p.3
  - @DOTPROD #1 p.3; #5 p.2
  - @IRR #4 p.2
  - @LABEL #3 p.6; #5 p.2
  - @LCHOOSE #5 p.2
  - @PMT #6 p.3
  - @PV #6 p.5
  - @RATE #4 p.4
  - @ROUND #3 p.6; #6 p. 3, 8
  - Control-K=command #4 p.7
  - Control-S=command #4 p.7
  - Keystroke Memory #4 p.7
  - Macros within Macros #4 p.9

SUBMITTING ARTICLES TO SATN

Articles and pointers submitted to SATN are considered for acceptance based on their quality and interest to SATN readers.

We would prefer to see queries first. If you submit an article without querying us first, include a stamped, self-addressed envelope (or mailer for diskettes). If you do not enclose an SSAE or mailer, we cannot return your material. Prospective authors may send for the *SATN Writer's Guide.*

Address all queries, articles, or requests for the guide to:

SATN
Editorial Department
P.O. Box 99
Newton, MA 02162